1. 题目
2. 题干

给定一个字符串 s 和一个非空字符串 p，找到 s 中所有是 p 的字母异位词的子串，返回这些子串的起始索引。

字符串只包含小写英文字母，并且字符串 s 和 p 的长度都不超过 20100。

说明：

字母异位词指字母相同，但排列不同的字符串。

不考虑答案输出的顺序。

1. 示例

输入:

s: "cbaebabacd" p: "abc"

输出:

[0, 6]

解释:

起始索引等于 0 的子串是 "cba", 它是 "abc" 的字母异位词。

起始索引等于 6 的子串是 "bac", 它是 "abc" 的字母异位词。

1. 题解
2. 思路
3. 暴力解法

字母异位词，即两个字符串中的字母都是相同的，就是顺序不同。因此，我们可以将p串中的每个字符进行hash计数，存入一个hashmap中。然后遍历字符串s，对每个p.length()长度的s子串，判断哪些子串是p的字母异位词。 对每个s的子串中的每个字符，判断是否出现在了p串中，如果出现了，则将hashmap中的对应字符出现次数相应地减掉1，如果某个字符没有出现或者多出现了1次，减掉1后hashmap中就会出现负数，说明该子串不是，break后继续遍历。这样就可以找到所有的子串了。

1. 滑动窗口 + 双指针

基本思路与暴力解法相同，首先都是要记录p串中字符的出现次数，然后用双指针作为滑动窗口的左右指针，初始化都指向0。创建一个临时计数数组tmp[]，记录滑动窗口内每个字符的出现次数，与p串中字符出现次数进行比较。右指针不断右移，每移入一个字符，将该字符的次数+1，如果该字符对应的数字大于p串中该字符的数字，则说明滑动窗口中该字符多了1个，则让左指针不断右移，将右移过程中碰到的字符的计数都-1，直到碰到一个该字符。这样，滑动窗口恢复正常，右指针可继续右移。若窗口的大小恰好为p串的长度，且每个字符的数字都没有超过p串的数字，说明滑动窗口内的字符串与p串是异位字符串，将窗口的左指针保存。

1. 代码实现
2. 暴力解法

Java：

class Solution {

    public List<Integer> findAnagrams(String s, String p) {

        //字符串 操作

        int sn = s.length();

        int pn = p.length();

        ArrayList<Integer> list = new ArrayList<>();

        if(s == null || p == null || sn < pn){

            return new ArrayList<Integer>();

        }

        // 由于字母只有26个，用数组保存出现次数即可

        int[] map = new int[26];

        for(int i = 0; i < pn; ++i){

            map[p.charAt(i) - 'a']++;

        }

        //遍历s，取子字符串

        for(int i = 0 ; i <= sn-pn; ++i){

            int[] tmp = Arrays.copyOf(map, 26);

            int j = i;

            for(; j < i+pn; ++j){

                if(--tmp[s.charAt(j)-'a'] < 0){

                    break;

                }

            }

            // 找到

            if(j == i+ pn){

                list.add(i);

            }

        }

        return list;

    }

}

1. 滑动窗口+ 双指针

Java：

// 滑动窗口 + 双指针

        // //字符串 操作

        int sn = s.length();

        int pn = p.length();

        ArrayList<Integer> list = new ArrayList<>();

        if(s == null || p == null || sn < pn){

            return new ArrayList<Integer>();

        }

        // 由于字母只有26个，用数组保存出现次数即可

        int[] map = new int[26];

        for(int i = 0; i < pn; ++i){

            map[p.charAt(i) - 'a']++;

        }

        //双指针

        int left = 0, right = 0;

        // 滑动窗口数组

        int[] win = new int[26];

        while(left < sn && right < sn){

            // 右指针为当前判断的字符

            int cur = s.charAt(right) - 'a';

            win[cur]++;

            right++;

            // 当右指针所指的字符次数超了，就右移左指针

            while(win[cur] > map[cur]){

                win[s.charAt(left)-'a']--;

                left++;

            }

            //如果滑动窗口与pn相等，说明找到一个

            if(right - left == pn){

                list.add(left);

            }

        }

        return list;